

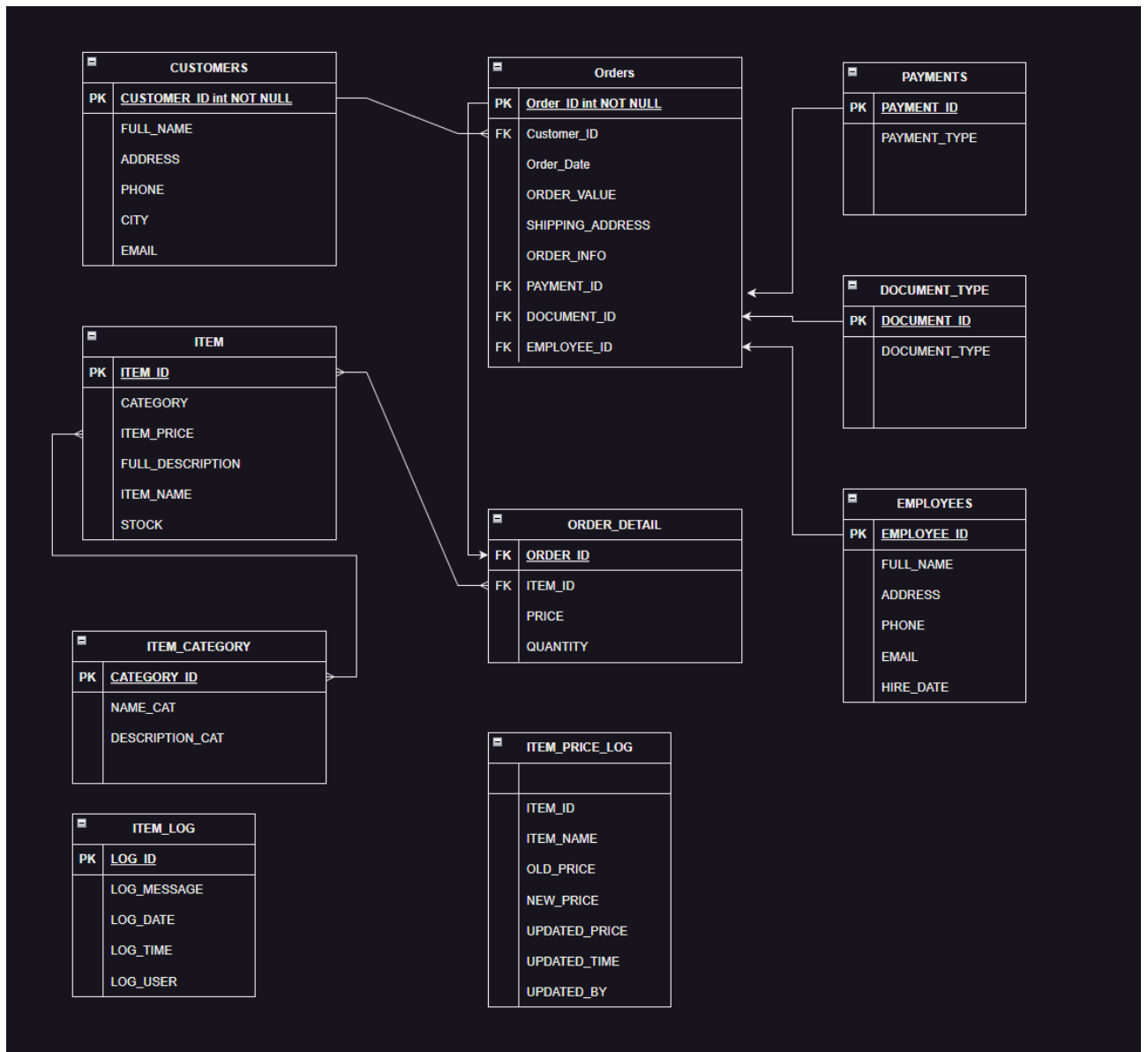
Proyecto Final

El modelo de negocio es para una pequeña empresa dedicada a la compra y venta de equipos tecnológicos de forma online E-Commerce (Laptops, consolas, desktops, monitores, etc). La cual en la actualidad se controla mediante una planilla Excel, donde dificulta encontrar información cuando se requiere hacerlo de forma rápida y eficiente.

El objetivo de este proyecto es poder controlar distintas cosas como por ejemplo el stock, ingreso de nuevos productos, eliminación de productos discontinuados, seguimiento a las ordenes de los clientes, entre otros. La idea es que todo sea lo mas simple posible, pero útil para las necesidades actuales de la empresa, logrando automatizar algunos de ellos, con posibilidad de añadir a futuro mas opciones y utilidades.

Como se mencionaba mas atrás la problemática a resolver es la lentitud con la qque se accede a los datos mediante la planilla Excel que se utiliza actualmente, cuando se requiere información de clientes, fechas de compra, garantías, entre otras. Se debe recurrir a la búsqueda tradicional con algún dato clave para acceder mas rápido a la información. Gracias al proyecto esto se logra al usar las primary key para llegar de forma rápida y obtener todos los resultados relacionados en un solo click.

Diagrama E-R



Enlace de archivo de diagrama

[https://github.com/Da3s/MySQL_Coder_Final/blob/main/Diagrama%20\(Laptops\).drawio](https://github.com/Da3s/MySQL_Coder_Final/blob/main/Diagrama%20(Laptops).drawio)

Enlace de Listado de tablas y descripción

https://github.com/Da3s/MySQL_Coder_Final/blob/main/Formato%20-%20Descripci%C3%B3n%20de%20tablas.xlsx

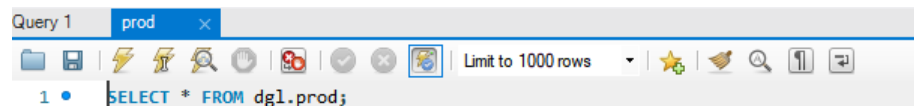
1. Vistas

Vista 1: prod

Descripción: Vista para ver productos disponibles ordenados por alfabeto, precio y stock

Objetivo: Poder revisar de manera rápida el stock y precio de determinado producto

Tablas/Datos: Se utiliza la tabla ITEM y los datos de columnas Item_name, Item_price y Stock



item_name	item_price	stock
Acer A314-22-R1L4-2	450000	1
Asus FA506QM-HN023T	950000	5
Asus FX506HC-HN042T	750000	44
Asus FX516PC-HN021T	850000	1
Asus G512LI-HN061T	750000	31
Asus G513QM-HF276T	1300000	24
Asus GA401QM-K2120T	1350000	93
Asus X409MA-EK173T	200000	30
Asus X413EA-EB667T	390000	85
Desk i5 10400	580000	9
Desk i7 10700	400000	80
Desk Ryzen 5	550000	41
Desktop i5 1660	650000	34
HP 14-DQ2026LA	300000	45
Lenovo Legion 5i	900000	6
Lenovo Legion 5i	850000	8
Monitor Gear 32p	200000	16
MSI G274	195990	50
Nintendo Switch	300000	4
PS5	650000	4
Switch Oled	350000	6
Xbox One S	270000	2

Vista 2: pends

Descripción: Vista para ver envíos pendientes o en proceso con datos de identificación y nombre

Objetivo: Permite revisar envíos que no han sido finalizados, para así proceder a enviar o terminar si ya fueron enviados al destinatario

Tablas/Datos: Se utiliza la tabla Orders y Customers, luego se llama a los datos de las tablas Customer_ID, Full_name y Order_info

Query 1 pends x

Limit to 1000 rows

```
1 • SELECT * FROM dg1.pends;
```

<

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [A](#)

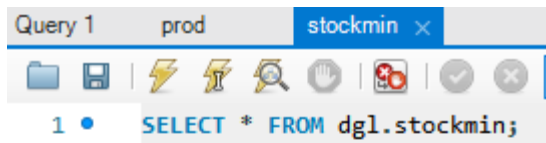
	customer_id	full_name	order_info
▶	21.304.686-1	Daniela Aylett Muñoz	PENDIENTE
	5.704.158-7	Miguel Antonio Silva	PENDIENTE
	5.625.957-0	Maximiliano Alfonso Perez	PROCESANDO
	20.759.844-5	Eleonora del Carmen Silva	PENDIENTE
	4.733.992-8	Juan Alberto San Martin	PROCESANDO
	11.610.044-4	Miguel Mauricio Fernandez	PROCESANDO
	19.195.252-9	Sofia Javiera Muñoz	PENDIENTE
	21.304.686-1	Daniela Aylett Muñoz	PENDIENTE

Vista 3: stockmin

Descripción: Vista para ver productos con stock bajo 10 unidades

Objetivo: Esta vista sirve para ver que productos tienen bajo stock y así poder reponer antes de que se agoten

Tablas/Datos: Se utiliza la tabla Item y los datos Item_id, Item_name y Stock



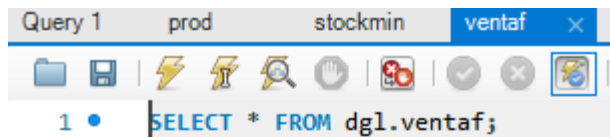
item_id	item_name	stock
335268	Asus FA506QM-HN023T	5
335295	Nintendo Switch	4
335349	Lenovo Legion 5i	6
335376	Asus FX516PC-HN021T	1
335457	Desk i5 10400	9
335565	Xbox One S	2
335592	PS5	4
335619	Switch Oled	6
335673	Lenovo Legion 5i	8
335700	Acer A314-22-R1L4-2	1

Vista 4: ventaf

Descripción: Vista para buscar ventas por fecha

Objetivo: Esta vista sirve para la entrega de boletas en caso de ser necesario reimprimir o para control

Tablas/Datos: Esta utiliza la talba Orders y los datos de Order_ID, Customer_ID y Order_Date



The image shows a database result grid with the following data:

	order_id	customer_id	order_date
▶	2	19.855.589-4	2021-09-08
	3	8.365.986-6	2021-09-08
	4	5.704.158-7	2021-09-08
	15	11.610.044-4	2021-09-08
	16	15.449.577-0	2021-09-08
	20	21.304.686-1	2021-09-08

Vista 5: totales

Descripción: Vista para ver ventas totales por id de producto

Objetivo: Poder ver las ventas totales por cada articulo y asi poder revisar que marcas y/o productos tienen mas salida

Tablas/Datos: Se utiliza la talba Order_detail y Item, los datos utilizados son Item_ID, Item_name, Price y Quantiy



Result Grid			
Filter Rows: <input type="text"/>			
Export:			
	item_id	item_name	Venta_total
▶	335295	Nintendo Switch	600000
	335511	Monitor Gear 32p	2200000
	335349	Lenovo Legion 5i	7200000
	335781	HP 14-DQ2026LA	1800000
	335754	Desktop i5 1660	2600000
	335484	Desk Ryzen 5	4400000
	335727	Desk i7 10700	800000
	335457	Desk i5 10400	3480000
	335808	Asus X409MA-EK173T	1600000
	335322	Asus G513QM-HF276T	1300000
	335403	Asus G512LI-HN061T	12750000
	335376	Asus FX516PC-HN021T	850000
	335430	Asus FX506HC-HN042T	18750000
	335268	Asus FA506QM-HN023T	3000000
	335700	Acer A314-22-R.1L4-2	1800000

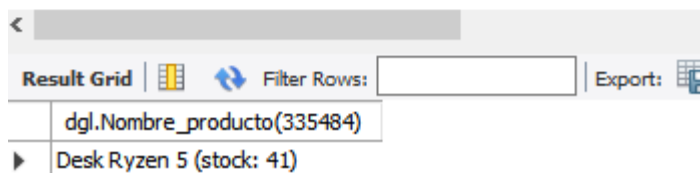
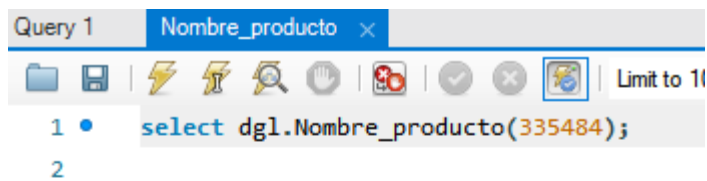
2. Funciones

Función 1: Nombre_producto

Descripción: Funcion que permite con el id de producto consultar nombre y stock

Objetivo: Poder acceder de forma rápida a la consulta del detalle de producto y su stock al ingresar el ID de producto

Tablas/Datos: Se utiliza la tabla Item y los datos de las columnas Item_name, Stock

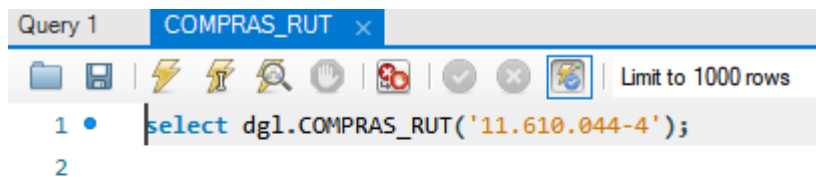


Función 2: COMPRAS_RUT

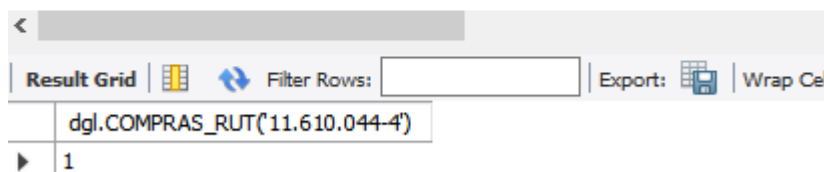
Descripción: Funcion que permite ver cuantas compras a realizado determinado rut o id de cliente

Objetivo: Saber por número de compras que clientes son los mas cativos y asi poder ofrecer descuentos

Tablas/Datos: Se utiliza la tabla Orders y los datos de la columna Customer_ID



```
Query 1  COMPRAS_RUT x
Limit to 1000 rows
1 • select dgl.COMPRAS_RUT('11.610.044-4');
2
```



Result Grid	Filter Rows:	Export:	Wrap Ce
dgl.COMPRAS_RUT('11.610.044-4')			
1			

3.

3. Stored Procedures

Procedure1: Direccion_envio

Descripción: Store Procedure que permite Obtener nombre de cliente y direccion para reparto de compra

Objetivo: Obtener de la BD rápidamente la dirección de envío para determinado cliente

Tablas/Datos: Se utiliza la tabla Customers y los datos de la columna Full_name y Address

Query 1 Direccion_envio x

Limit to 1000 rows

```
1 • call dgl.Direccion_envio('20.759.844-5');
2
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cor

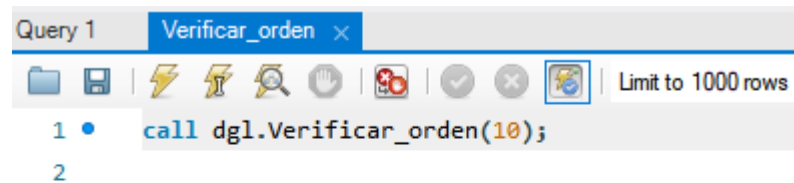
FULL_NAME	ADDRESS
Eleonora del Carmen Silva	Eleuterio Ramirez 56

Procedure 2: Verificar_orden

Descripción: Store Procedure que permite estado de orden

Objetivo: Poder revisar rápidamente al ingresar Numero de orden, el detalle de esta orden

Tablas/Datos: Se utilizan las tablas de Order_detail y Orders, junto con los datos de las columnas Item_ID, Price, Quantity y Order_info



Query 1 Verificar_orden x

Limit to 1000 rows

```
1 • call dgl.Verificar_orden(10);
2
```



Result Grid | Filter Rows: | Export: | Wrap Cell Co

item_id	price	quantity	order_info
335457	580000	6	ENTREGADO

4. Triggers

4.1 Trigger 1:

Descripción: Descuenta Stock al ingresar un nuevo registro en ORDER y Luego en ORDER_DETAIL

Objetivo: Llevar stock en tiempo real al realizar una venta, descontando al instante las unidades vendidas

Tablas/Datos: Se activa antes de insertar una nueva orden en Order_detail

4.2 Trigger 2:

Descripción: Registra quien actualizo el precio del producto con detalle de fecha

Objetivo: Llevar control de los cambios de precio

Tablas/Datos: Se aplica despues de actualizar el valor de un producto en la tabla item y registra en la tabla Item_Price_Log los cambios, utilizando las columnas Item_ID, OLD_price_New_price, Updated_Date, Updated_Time y Updated_By para tener un registro lo mas detallado posible

Query 1

```

237  FOR EACH ROW
238  BEGIN
239      IF NEW.ITEM_PRICE <> OLD.ITEM_PRICE THEN
240          INSERT INTO ITEM_PRICE_LOG (ITEM_ID, ITEM_NAME, OLD_PRICE, NEW_PRICE, UPDAT
241          VALUES (NEW.ITEM_ID, OLD.ITEM_NAME, OLD.ITEM_PRICE, NEW.ITEM_PRICE, CURDATE
242      END IF;
243  END;
244  //
245
246  /*Llamo a la tabla para verificar que registro el cambio*/
247  SELECT * FROM ITEM_PRICE_LOG;
248

```

Limit to 1000 rows

Result Grid

	ITEM_ID	ITEM_NAME	OLD_PRICE	NEW_PRICE	UPDATED_DATE	UPDATED_TIME	UPDATED_BY
▶	335268	Asus FA506QM-HN023T	1000000	950000	2023-05-09	22:11:08	encargado
	335895	MSI G274	199990	195990	2023-05-09	22:47:14	root@localhost

4.3 Trigger 3:

Descripción: Registra en el log, los ingresos de nuevos productos, junto con fecha de ingreso

Objetivo: Para llevar un registro al añadir nuevos productos y en que fecha estos llegaron

Tablas/Datos: Se gatilla después de Insertar el Nuevo producto en la tabla Item, modificand en detalle luego en la tabla Item_Log este cambio

Query 1

```

276     SET LOG_USER = USER();
277     INSERT INTO ITEM_LOG (LOG_MESSAGE, LOG_DATE, LOG_TIME, LOG_USER) VALUES (log_me
278 END;
279 $$
280
281 /*Para insertar nuevo prodcuto*/
282 INSERT INTO ITEM (ITEM_ID, CATEGORY, ITEM_PRICE, FULL_DESCRIPTION, ITEM_NAME, STOCK
283 VALUES (335985, 4, 199990, 'Monitor Gamer', 'MSI G297', 50);
284
285 /*Llamamos a la tabla item log para verificar que registro correctamente*/
286 select * from item_log;
287

```

Result Grid

	LOG_ID	LOG_MESSAGE	LOG_DATE	LOG_TIME	LOG_USER
▶	1	Se ingresó 50 unidades del producto MSI G297 (ID: 335985).	2023-05-09	22:53:46	root@localhost
	2	Se eliminó el producto MSI G297 (ID: 335985).	2023-05-09	22:55:04	root@localhost
*	NULL	NULL	NULL	NULL	NULL

4.4 Trigger 4:

Descripción: Para registrar en el log cuando se elimina un producto descontinuado

Objetivo: Poder identificar en detalle productos eliminados de la BD y controlar quien lo realizo

Tablas/Datos: Se gatilla Antes de eliminar el producto de la tabla ITEM, luego queda todo registrado en detalle en la Tabla Item_LOG

Query 1

```

276 SET LOG_USER = USER();
277 INSERT INTO ITEM_LOG (LOG_MESSAGE, LOG_DATE, LOG_TIME, LOG_USER) VALUES (log_me
278 END;
279 $$
280
281 /*Para insertar nuevo prodcto*/
282 INSERT INTO ITEM (ITEM_ID, CATEGORY, ITEM_PRICE, FULL_DESCRIPTION, ITEM_NAME, STOCK
283 VALUES (335985, 4, 199990, 'Monitor Gamer', 'MSI G297', 50);
284
285 /*Llamamos a la tabla item log para verificar que registro correctamente*/
286 select * from item_log;
287

```

Result Grid

	LOG_ID	LOG_MESSAGE	LOG_DATE	LOG_TIME	LOG_USER
▶	1	Se ingresó 50 unidades del producto MSI G297 (ID: 335985).	2023-05-09	22:53:46	root@localhost
	2	Se eliminó el producto MSI G297 (ID: 335985).	2023-05-09	22:55:04	root@localhost
*	NULL	NULL	NULL	NULL	NULL

5. Archivos SQL

5.1 Script completo de la BD

https://github.com/Da3s/MySQL_Coder_Final/blob/main/Script%20entrega%20final%20Coder.sql